

REMARKS

Claims 1-38 remain pending in the present application. Applicant amends independent Claims 1, 7, 15, 24, and 35 to clarify claimed subject matter and/or correct informalities. The original specification and drawings support these claim amendments at least at pages 2, 9, and in Figures 3, 5, and 6. Therefore, these revisions introduce no new matter.

Claims 1-38 are for consideration upon entry of the present Amendment. Applicant requests favorable reconsideration of this response and allowance of the subject application based on the following remarks.

Previous Claims Rejections Under 35 USC § 101 and §103

Applicant appreciates Examiner's withdrawal of the 35 U.S.C. §101 and §103 rejections from the previous Office Action.

Claim Rejections 35 U.S.C. §101

Claims 7-23 stand rejected under 35 U.S.C. § 101 as being directed to non-statutory subject matter.

Without conceding the propriety of the stated rejections, and only to advance the prosecution of this application, Applicant amends **independent Claim 7**, to clarify further features of the subject matter. Claim 7 recites in part, "data structure stored on one or more computer-readable media having computer executable instructions stored on a computing device". Support may be found in the original specification. Thus, no new matter has been introduced.

Without conceding the propriety of the stated rejections, and only to advance the prosecution of this application, Applicant amends **independent Claim 15**, to clarify further features of the subject matter. Claim 15 recites in part, “a memory; a processor coupled to the memory for executing a query evaluation”. Support may be found in the original specification. Thus, no new matter has been introduced.

Dependent Claims 8-14 and 16-23 depend directly or indirectly from one of independent Claims 7 and 15, respectively, and thus are allowable as depending from an allowable base claim.

These claims comply with 35 U.S.C. §101 and as a result, the rejection is now moot. Applicant respectfully requests that the rejection of Claims 7-23 be withdrawn.

Claim Rejections under 35 U.S.C. § 103

Claims 1-38 stand rejected under 35 U.S.C. § 103(a) as being obvious over non-patent literature titled “Efficient Filtering of XML Documents for Selective Dissemination of Information” by Mehmet Altinel, et al., 26th VLDB Conference, 2000, pages 53-64 (previously presented and known hereinafter as “Altinel”) in view of non-patent literature titled “On Efficient Matching of Streaming XML Documents and Queries” by Sailaja et al, University of British Columbia, Canada, 2002, pages 1-20 (hereinafter “Sailaja”), and further in view of non-patent literature titled “Index Structures for Selective Dissemination of Information Under the Boolean Model” by Yan et al., ACM Transactions on Database Systems, vol. 19, No. 2, June 1994, pages 332-364 (hereinafter “Yan”). Applicant respectfully traverses the rejection.

Without conceding the propriety of the stated rejections, and only to advance the prosecution of this application, Applicant amends **independent Claim 1**, to clarify further features of the subject matter. Amended Claim 1 now recites a method, comprising:

- receiving an input, wherein the input comprises elemental language units;
- evaluating the input against multiple queries by evaluating common query expressions of the multiple queries in parallel, at a same time;**
- generating at least some of the elemental language units into opcodes;
- merging opcodes into an opcode tree, wherein no opcodes are added to the opcode tree during an active merging,**
 - wherein the language units have been parsed and compiled into opcodes;
- traversing the opcode tree of hierarchical nature that includes a plurality of opcode nodes which together define opcodes that should be executed to evaluate a plurality of queries;
- executing each of the opcode nodes in the opcode tree as each opcode node is encountered in the traversal to evaluate the plurality of queries against the input;
- indexing branch opcodes to provide a framework for insertion of indexing techniques that are customized to a type of comparison;
- maintaining the opcode tree that is used during processing by making a copy of the opcode tree; and**
 - updating the opcode tree copy, **wherein the opcode nodes are removed from the opcode tree while the opcode tree copy is used for query processing;**
 - wherein a relationship between the opcodes and the opcode tree is embedded in the opcodes that is created when a query is compiled.

Applicant respectfully submits that Altinel, Sailaja, and/or Yan, alone or in combination, fail to disclose, teach, or suggest such a method.

References Fail to Teach or Suggest Evaluating Multiple Queries, No Opcodes Added During Merging, and No Removing of Opcodes

First, Altinel fails to teach or suggest the features in Applicant's claims. Applicant agrees with the Office that Altinel fails to explicitly teach the features of Claim 1 (Office

Action, pgs. 5-6). Rather, Altinel describes index organizations and search algorithms for performing efficient filtering of XML documents for large-scale information dissemination systems (Abstract). Altinel is a document filtering system, named XFilter, that provides highly efficient matching of XML documents to large numbers of user profiles (page 53, col. 2, 3rd paragraph). In particular, Altinel mentions XFilter **examines one document at a time**, so only path expressions over individual documents are needed (page 55, col. 1, 2nd paragraph). In contrast, Applicant's amended Claim 1 recites in part, "*evaluating the input against multiple queries by evaluating common query expressions of the multiple queries in parallel, at a same time*".

Second, Sailja fails to compensate for the deficiencies of Altinel. Sailja describes efficient matching of requirements to specifications (Abstract). There are three lists in Sailja, each associated with data tree node; Query Labeling (QL) list, which contains queries answered by the node, Chain Matching List (CML) that tracks queries that have matched, and a Push List, an auxiliary list to manage CML (Section 4). These lists are not making copies of the opcode tree. Nowhere is there any discussion or even mention in Sailja of "*maintaining the opcode tree copy that is used during processing by making a copy of the opcode tree*", as recited in Applicant's Claim 1. Thus, Sailja does not provide what is missing from Altinel to support a §103 rejection.

Third, Applicant agrees with the Office that the combination of Altinel and Sailja do not teach the method of merging opcodes into an opcode tree, wherein the language units have been parsed and compiled into opcodes; indexing branch opcodes to provide a framework for insertion of indexing techniques that are customized to a type of comparison (Office Action, pg. 7). Yan is directed towards selective dissemination of information, where

profiles, i.e., queries that are constantly evaluated will be automatically informed of new additions (Abstract). While Yan mentions the word “merging”, it is used in the context of merging (union) the lists (pg. 335), not “*merging opcodes into an opcode tree, wherein the language units have been parsed and compiled into opcodes*”, as recited in Applicant’s amended Claim 1. The list in Yan contains profiles, a document is checked against an index of profiles. Thus, Yan fails to disclose, teach, or suggest “*merging opcodes into an opcode tree, wherein no opcodes are added to the opcode tree during an active merging*”, as recited in Applicant’s Claim 1.

Altinel, Sailaja, and Yan, alone or in combination, fail to disclose, teach or suggest “*evaluating the input against multiple queries by evaluating common query expressions of the multiple queries in parallel, at a same time; merging opcodes into an opcode tree, wherein no opcodes are added to the opcode tree during an active merging, updating the opcode tree copy, wherein the opcode nodes are removed from the opcode tree while the opcode tree copy is used for query processing*”, as recited in Applicant’s amended Claim 1. Accordingly, Applicant submits that the evidence relied upon by the Office no longer supports the rejections made under §103(a).

Insufficient Evidence to Suggest Reason to Modify References

Fourth, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness... KSR Int’l Corp. v. Teleflex, Inc., Slip Op. at 14 (U.S. Apr. 30, 20076) (quoting In re Kahn, 441 F.3d 977, 988 (CA Fed. 2006)). The Office stated it would have been obvious to a person of ordinary skill to modify the teachings of Altinel with the teachings of Sailaja and with the further teachings of Yan to include the

method of merging opcodes into an opcode tree, wherein the language units have been parsed and compiled into opcodes; indexing branch opcodes to provide a framework for insertion of indexing techniques that are customized to a type of comparison with the motivation to perform efficient filtering of XML documents for large-scale information dissemination systems (Office Action, pg. 8). Applicant respectfully disagrees and submits that this modification is not well reasoned, because there is nothing in either of the references that would suggest this reason.

Furthermore, there is no articulated reason with some rational underpinning to support this rejection. Instead, the asserted reason relies on hindsight without evidence of articulated reasoning to propose the suggested modification. This rejection is improper for this additional reason.

Independent Claims 7, 15, 24, and 35 are directed to a computer-readable media, a system, a computer-readable storage media, and a method, respectively, and each is allowable for reasons similar to those discussed above with respect to Claim 1.

Independent Claim 7 recites: “an opcode tree data structure stored on one or more computer-readable media having computer executable instructions stored on a computing device, comprising a plurality of hierarchical opcode nodes that represent a plurality of opcodes that are executed as each opcode node is encountered to evaluate a set of queries represented by the opcode tree, wherein the opcode tree that is used during processing is copied and updated; the instructions further comprising to evaluate an input against multiple queries by evaluating common query expressions of multiple queries in parallel at a same time; and the instructions further comprising to update an opcode tree copy, wherein the plurality of opcode nodes are removed from the opcode tree while the opcode tree copy is

used for query processing” as recited in Applicant’s amended Claim 7. Applicant respectfully submits that Altinel, Sailaja, and Yan, alone or in combination, fail to disclose, teach, or suggest these features as recited in Claim 7.

Independent Claim 15 recites a query evaluation system, comprising:

- memory;
- a processor coupled to the memory for executing a query evaluation with elemental language;
- a language analysis module generating elemental language input into opcodes, wherein input against multiple queries is evaluated by evaluating common query expressions of the multiple queries in parallel at a same time;;
- an opcode merger configured to combine opcodes that are derived from compiling expressions into an opcode tree, wherein the opcode merger detects using an optimization algorithm and combines literal comparisons into an indexed literal branch opcodes, wherein no opcodes are added to the opcode tree during an active merging;
- the opcode tree of hierarchical nature stored in memory and containing opcode nodes that include opcode objects corresponding to a plurality of queries, each opcode object that is common to multiple queries being represented by a single opcode node;
- a query processor configured to execute each opcode node as encountered of the opcode tree one time to evaluate the plurality of queries; and
- the opcode tree that is used during processing by the query processor is copied and updated, wherein the opcode nodes are removed from the opcode tree while the opcode tree copy is used for query processing;
- wherein a relationship between the opcodes and the opcode tree is embedded in the opcodes that is created when a query is compiled.

Applicant respectfully submits that Altinel, Sailaja, and Yan, alone or in combination, fail to disclose, teach, or suggest these features as recited in Claim 15.

Independent Claim 24 recites one or more computer-readable storage media containing computer-executable instructions that, when executed by a computer, perform the following steps:

- evaluating input against multiple queries by evaluating common query expressions of the multiple queries in parallel at a same time;

generating an input of elemental language units into opcodes;
merging opcodes into an opcode tree, wherein the language units have been parsed and compiled into opcodes, wherein no opcodes are added to the opcode tree during an active merging;

executing opcode nodes as encountered in the opcode tree of hierarchical nature to evaluate a plurality of queries represented in the opcode tree, at least one opcode node corresponding to at least a portion of two or more of the plurality of queries;

indexing branch opcodes to provide a framework for insertion of indexing techniques that are customized to a type of comparison;

caching an execution context derived from the execution of a first segment of opcode nodes;

re-using the execution context when executing a second opcode node;

maintaining the opcode tree that is used during processing by making a copy of the opcode tree; and

updating the opcode tree copy, wherein the opcode nodes are removed from the opcode tree while the opcode tree copy is used for query processing;

wherein a relationship between the opcodes and the opcode tree is embedded in the opcodes that is created when a query is compiled.

Applicant respectfully submits that Altinel, Sailaja, and Yan, alone or in combination,

fail to disclose, teach, or suggest these features as recited in Claim 24.

Independent Claim 35 recites a method, comprising:

evaluating input against multiple queries by evaluating common query expressions of the multiple queries in parallel at a same time;

merging opcodes into an opcode tree, wherein elemental language units have been parsed and compiled into opcodes, wherein no opcodes are added to the opcode tree during an active merging;

executing an opcode tree of hierarchical nature that includes opcode nodes as encountered that each correspond to one or more of a plurality of XPath queries represented by the opcode nodes, at least a first opcode node corresponding to a first query and a second query;

indexing branch opcodes to provide a framework for insertion of indexing techniques that are customized to the type of comparison;

using interim values from an execution context created in the execution of the first opcode node in the execution of a second opcode node corresponding to the second query to avoid re-creating at least a portion of the execution context;

maintaining the opcode tree that is used during processing by making a copy of the opcode tree; and

updating the opcode tree copy, wherein the opcode nodes are removed from the opcode tree while the opcode tree copy is used for query processing;

wherein a relationship between the opcodes and the opcode trees is embedded in the opcodes that is created when a query is compiled.

Applicant respectfully submits that Altinel, Sailaja, and Yan, alone or in combination, fail to disclose, teach, or suggest these features as recited in Claim 35.

Dependent Claims 2-6, 8-14, 16-23, 25-34, and 36-38 depend directly or indirectly from one of independent Claims 1, 7, 15, 24, and 35, respectively, and are allowable by virtue of this dependency. These claims are also allowable for their own recited features that, in combination with those recited in Claims 1, 7, 15, 24, and 35 are not disclosed, taught, or suggested by Altinel, Sailaja, or Yan, alone or in combination.

Applicant respectfully submits that Altinel, Sailaja, or Yan,, alone or in combination, do not render the claimed subject matter obvious and that the claimed subject matter, therefore, patentably distinguishes over the cited references. For all of these reasons, Applicant respectfully request the §103(a) rejection of these claims be withdrawn.

Conclusion

Claims 1-38 are in condition for allowance. Applicant respectfully requests reconsideration and prompt allowance of the subject application. If any issue remains unresolved that would prevent allowance of this case, the Office is requested to contact the undersigned attorney to resolve the issue.

Respectfully Submitted,

Lee & Hayes, PLLC
421 W. Riverside Avenue, Suite 500
Spokane, WA 99201

Dated: 4-14-2008

By: /Shirley Lee Anderson/
Shirley Lee Anderson
Reg. No. 57,763
509.324.9256 ext. 258